of some of the inventions since the mid-1990s, resulting in a simplification of pipelines in the more recent versions of microarchitectures.

To sustain the advances in processing performance via parallel processors, Amdahl's law suggests that another part of the system will become the bottleneck. That bottleneck is the topic of the next chapter: the memory system.

4.15 Historical Perspective and Further Reading

This section, which appears on the CD, discusses the history of the first pipelined processors, the earliest superscalars, and the development of out-of-order and speculative techniques, as well as important developments in the accompanying compiler technology.

4.16 Exercises

Contributed by Milos Prvulovic of Georgia Tech

Exercise 4.1

Different instructions utilize different hardware blocks in the basic single-cycle implementation. The next three problems in this exercise refer to the following instruction:

	Instruction	Interpretation
a.	`AND Rd,Rs,Rt`	`Reg[Rd]=Reg[Rs] AND Reg[Rt]`
b.	`SW Rt,Offs(Rs)`	`Mem[Reg[Rs]+Offs]=Reg[Rt]`

4.1.1 [5] <4.1> What are the values of control signals generated by the control in Figure 4.2 for this instruction?

4.1.2 [5] <4.1> Which resources (blocks) perform a useful function for this instruction?

4.1.3 [10] <4.1> Which resources (blocks) produce outputs, but their outputs are not used for this instruction? Which resources produce no outputs for this instruction?

Different execution units and blocks of digital logic have different latencies (time needed to do their work). In Figure 4.2 there are seven kinds of major blocks. Latencies of blocks along the critical (longest-latency) path for an instruction determine the minimum latency of that instruction. For the remaining three problems in this exercise, assume the following resource latencies:

	I-Mem	Add	Mux	ALU	Regs	D-Mem	Control
a.	200ps	70ps	20ps	90ps	90ps	250ps	40ps
b.	750ps	200ps	50ps	250ps	300ps	500ps	300ps

4.1.4 [5] <4.1> What is the critical path for an MIPS AND instruction?

4.1.5 [5] <4.1> What is the critical path for an MIPS load (LD) instruction?

4.1.6 [10] <4.1> What is the critical path for an MIPS BEQ instruction?

Exercise 4.2

The basic single-cycle MIPS implementation in Figure 4.2 can only implement some instructions. New instructions can be added to an existing ISA, but the decision whether or not to do that depends, among other things, on the cost and complexity such an addition introduces into the processor datapath and control. The first three problems in this exercise refer to this new instruction:

	Instruction	Interpretation
a.	`SEQ Rd,Rs,Rt`	`Reg[Rd] = Boolean value (0 or 1) of (Reg[Rs] == Reg[Rs])`
b.	`LWI Rt,Rd(Rs)`	`Reg[Rt] = Mem[Reg[Rd]+Reg[Rs]]`

4.2.1 [10] <4.1> Which existing blocks (if any) can be used for this instruction?

4.2.2 [10] <4.1> Which new functional blocks (if any) do we need for this instruction?

4.2.3 [10] <4.1> What new signals do we need (if any) from the control unit to support this instruction?

When processor designers consider a possible improvement to the processor datapath, the decision usually depends on the cost/performance trade-off. In the following three problems, assume that we are starting with a datapath from Figure 4.2, where I-Mem, Add, Mux, ALU, Regs, D-Mem, and Control blocks have latencies of 400ps, 100ps, 30ps, 120ps, 200ps, 350ps, and 100ps, respectively, and costs of 1000, 30, 10, 100, 200, 2000, and 500, respectively. The remaining three problems in this exercise refer to the following processor improvement:

	Improvement
a.	Add Multiplier to ALU
b.	Simpler Control

4.2.4 [10] <4.1> V

4.2.5 [10] <4.1> V

4.2.6 [10] <4.1> improvement.

Exercise 4.3

Problems in this exe

a.	Small Multiplexor (
b.	Small 8-bit ALU th

4.3.1 [5] <4.1, 4.2

4.3.2 [20] <4.1, 4. OR, NOT, and D Fli

4.3.3 [10] <4.1, 4. must all be 2-input

Cost and latency o (gates) that are avail problems in this exe

	NOT	
	Latency	Cos
a.	10ps	2
b.	20ps	2

4.8.6 [40] <4.3, 4.4> Using a single test described in 4.8.1, we can test for faults in several different signals, but typically not all of them. Describe a series of tests to look for this fault in all Mux outputs (every output bit from each of the five Muxes). Try to do this with as few single-instruction tests as possible.

Exercise 4.9

In this exercise we examine the operation of the single-cycle datapath for a particular instruction. Problems in this exercise refer to the following MIPS instruction:

	Instruction
a.	`SW R4,-100(R16)`
b.	`SLT R1,R2,R3`

4.9.1 [10] <4.4> What is the value of the instruction word?

4.9.2 [10] <4.4> What is the register number supplied to the register file's "Read register 1" input? Is this register actually read? How about "Read register 2"?

4.9.3 [10] <4.4> What is the register number supplied to the register file's "Write register" input? Is this register actually written?

Different instructions require different control signals to be asserted in the datapath. The remaining problems in this exercise refer to the following two control signals from Figure 4.24:

	Control Signal 1	Control Signal 2
a.	ALUSrc	Branch
b.	Jump	RegDst

4.9.4 [20] <4.4> What is the value of these two signals for this instruction?

4.9.5 [20] <4.4> For the datapath from Figure 4.24, draw the logic diagram for the part of the control unit that implements just the first signal. Assume that we only need to support `LW`, `SW`, `BEQ`, `ADD`, and `J` (jump) instructions.

4.9.6 [20] <4.4> Repeat 4.9.5, but now implement both of these signals.

Exercise 4.11

In this exercise we examine in detail how an instruction is executed in a single-cycle datapath. Problems in this exercise refer to a clock cycle in which the processor fetches the following instruction word:

	Instruction word
a.	10101100011000100000000000010100
b.	00000000100000100000100000101010

4.11.1 [5] <4.4> What are the outputs of the sign-extend and the jump "Shift left 2" unit (near the top of Figure 4.24) for this instruction word?

4.11.2 [10] <4.4> What are the values of the ALU control unit's inputs for this instruction?

4.11.3 [10] <4.4> What is the new PC address after this instruction is executed? Highlight the path through which this value is determined.

The remaining problems in this exercise assume that data memory is all zeros and that the processor's registers have the following values at the beginning of the cycle in which the above instruction word is fetched:

	R0	R1	R2	R3	R4	R5	R6	R8	R12	R31
a.	0	–1	2	–3	–4	10	6	8	2	–16
b.	0	256	–128	19	–32	13	–6	–1	16	–2

4.11.4 [10] <4.4> For each Mux, show the values of its data output during the execution of this instruction and these register values.

4.11.5 [10] <4.4> For the ALU and the two add units, what are their data input values?

4.11.6 [10] <4.4> What are the values of all inputs for the "Registers" unit?

Exercise 4.12

In this exercise, we examine how pipelining affects the clock cycle time of the processor. Problems in this exercise assume that individual stages of the datapath have the following latencies:

	IF	ID	EX	MEM	WB
a.	250ps	350ps	150ps	300ps	200ps
b.	200ps	170ps	220ps	210ps	150ps

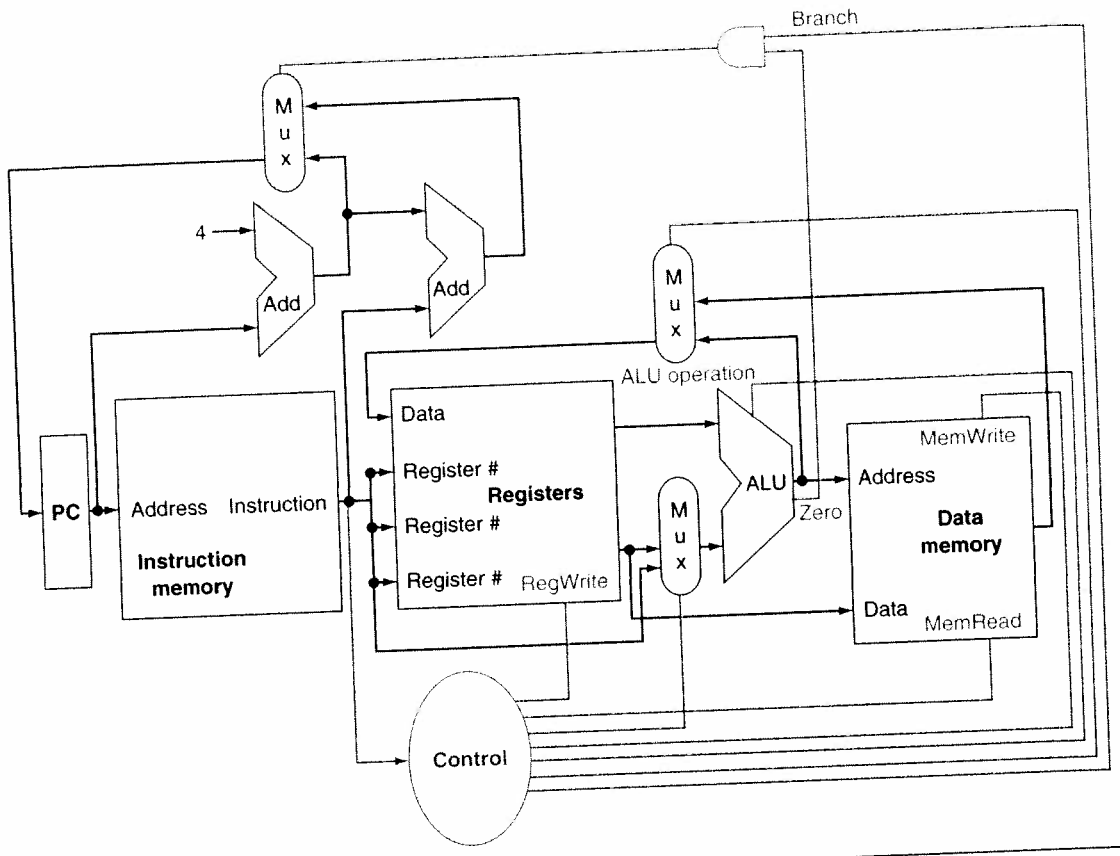

FIGURE 4.2 The basic implementation of the MIPS subset, including the necessary multiplexors and control lines. The top multiplexor ("Mux") controls what value replaces the PC (PC + 4 or the branch destination address); the multiplexor is controlled by the gate that "ANDs" together the Zero output of the ALU and a control signal that indicates that the instruction is a branch. The middle multiplexor, whose output returns to the register file, is used to steer the output of the ALU (in the case of an arithmetic-logical instruction) or the output of the data memory (in the case of a load) for writing into the register file. Finally, the bottommost multiplexor is used to determine whether the second ALU input is from the registers (for an arithmetic-logical instruction OR a branch) or from the offset field of the instruction (for a load or store). The added control lines are straightforward and determine the operation performed at the ALU, whether the data memory should read or write, and whether the registers should perform a write operation. The control lines are shown in color to make them easier to see.

you have little or no background in digital logic, you will find it helpful to read Appendix C before continuing.

The datapath elements in the MIPS implementation consist of two different types of logic elements: elements that operate on data values and elements that contain state. The elements that operate on data values are all combinational, which means that their outputs depend only on the current inputs. Given the same input, a combinational element always produces the same output. The ALU shown in Figure 4.1 and discussed in Appendix C is an example of a combinational

combinational element An operational element, such as an AND gate or an ALU.

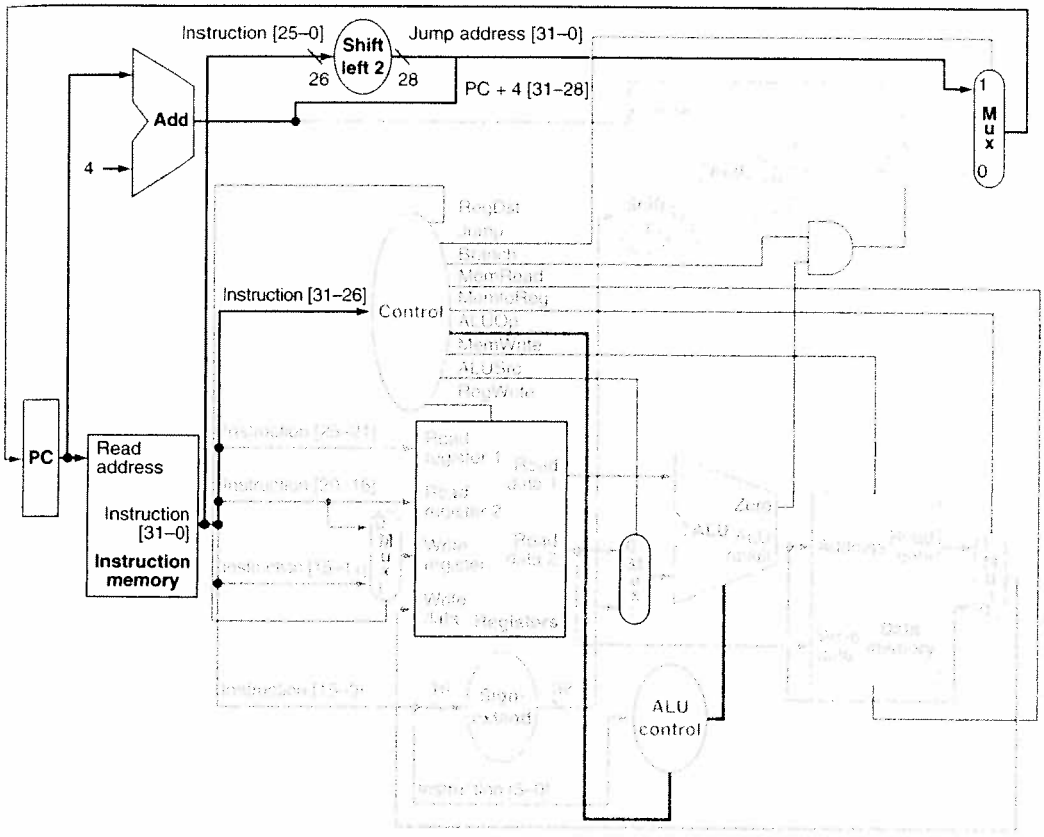

FIGURE 4.24 The simple control and datapath are extended to handle the jump instruction. An additional multiplexor (at the upper right) is used to choose between the jump target and either the branch target or the sequential instruction following this one. This multiplexor is controlled by the jump control signal. The jump target address is obtained by shifting the lower 26 bits of the jump instruction left 2 bits, effectively adding 00 as the low-order bits, and then concatenating the upper 4 bits of PC + 4 as the high-order bits, thus yielding a 32-bit address.

the clock cycle is determined by the longest possible path in the processor. This path is almost certainly a load instruction, which uses five functional units in series: the instruction memory, the register file, the ALU, the data memory, and the register file. Although the CPI is 1 (see Chapter 1), the overall performance of a single-cycle implementation is likely to be poor, since the clock cycle is too long.